REMARKS

Reconsideration of the above-identified patent application in view of the amendments above and the remarks following is respectfully requested.

Claims 1 and 3-35 are in this case. Claims 22 and 23 have been rejected under § 102(b). Claims 1, 3-21 and 24-35 have been rejected under § 103(a). Claims 31-35 have been canceled. New claim 36 has been added.

The claims before the Examiner are directed toward a flash memory device for storing code to be executed by a processor. The device includes a flash memory, for storing the code, such that the code cannot be executed in place from the flash memory. The device also includes a volatile memory to which the code is copied for execution, a logic, separate from the processor, for doing the copying, and a bus via which the flash memory, the volatile memory and the logic communicate directly with each other. If the code is boot code, the device can be used to boot a system, that includes the processor, in response to a power-on signal.

§ 102(b) Rejections – Garfunkel et al. '404

The Examiner has rejected claims 22 and 23 under § 102(b) as being anticipated by Garfunkel et al., US Patent No. 6,615,404 (henceforth, "Garfunkel et al. '404"). The Examiner's rejection is respectfully traversed.

Garfunkel et al. '404 teach an embedded system **10** and a method of upgrading the software of the system. Embedded system **10** includes a controller **11**, a flash memory **12** and a RAM **13**. The current boot code of embedded system **10** is stored in an ordinary sector **21** of flash memory **12**. The original boot code of embedded system **10** is stored in a special sector **20** of flash memory **12** that is hardware-protected, for example by requiring an unusually high voltage for reprogramming. If

an updating of the system software in flash memory **12** is unexpectedly interrupted, the original boot code is available as a backup in sector **20** to re-start the system.

The primary difference between the teachings of Garfunkel et al. '404 and the present invention, as recited in independent claim 22, is that claim 22 requires the flash memory to be such that code can not be directly executed therefrom. It is clear form Garfunkel et al. '404 that flash memory **12** is executable. See, for example, column 6 line 66 through column 7 line 3:

> According to another preferred embodiment of the invention, one or more sectors **22** of the flash memory **12** are also copied into the RAM **13**. In this case, the controller **11** operates that section of the program from the RAM **13**, <u>rather than from the flash memory **12**</u>. (emphasis added).

In other words, during normal operation of embedded system **10**, as opposed to when the code in sectors **22** of flash memory **12** is being upgraded, controller **11** has the option of executing the code directly from those sectors **22**. The Examiner misinterpreted the subsequent lines (column 7 lines 5-8) in Garfunkel et al. '404,

> In current practice, during updating, software can not be run from the flash memory **12** at all, and therefore, only software copied into RAM **13** is functional.

as indicating that flash memory **12** is not directly executable. The operative word in this passage is "during updating". The point of this passage is that in order to update the code in flash memory **12**, the old code must be erased, and so is not available for execution. During <u>normal</u> operation of the system, as opposed to during an <u>update</u> of the code in flash memory **12**, the code in flash memory **12** <u>is</u> directly executable. Thus, the present invention, as recited in independent claim 22, is not anticipated by Garfunkel et al. '404.

Furthermore, the present invention, as recited in independent claim 22, is not even obvious from Garfunkel et al. '404. It is not obvious from Garfunkel et al. '404

that a non-executable flash memory can be used to store boot code. As best understood, during normal operation (as opposed to immediately following an interrupted software upgrade), the boot code of Garfunkel et al. '404 must be executed directly from sector **21** of flash memory **12** because RAM **13**, being volatile, is blank when the system is powered up. As stated in column 4 lines 44-46,

> The flash memory **12** is the only non-volatile memory which is required for storing the operating software and/or other desired instructions. (emphasis added)

In other words, no other non-volatile memory need be available for executing boot code during normal system power-up. Therefore, flash memory **12** must be executable.

The present invention gets around this problem by including logic **42** to copy boot code from non-executable flash memory **14** to executable SRAM **40** when system **30** of the present invention powers up. There is neither a hint nor a suggestion of special circuitry analogous to logic **42**, separate from controller **11**, in embedded system **10** of Garfunkel et al. '404 for copying boot code from sector **21** of flash memory **12** to RAM **13** on power-up.

With independent claim 22 allowable in its present form, it follows that claim 23, that depends therefrom, also is allowable.

§ 103(a) Rejections – Brown et al. '739 and Kakinuma et al. '349 in view of Garner '482

The Examiner has rejected claims 1, 10-13, 16-18, 20, 21, 29 and 30 under § 103(a) as being unpatentable over Brown et al., US Patent No. 6,201,739 (henceforth, "Brown et al. '739") and Kakinuma et al., US Patent No. 5,640,349 (henceforth, "Kakinuma et al. '349") and further in view of Garner, US Patent No.

6,549,482 (henceforth, "Garner '482"). The Examiner's rejection is respectfully traversed.

Brown et al. '739 teach a flash memory device with a suspend pin for suspending write and erase operations to allow a read operation to take priority over a write or erase operation already in progress. The problem addressed by their device, as described in column 3 lines 36-60, is that of simultaneous management of data storage and direct execution of code on the same flash memory: data storage occasionally needs write and erase operations, which interfere with the immediate access to code that is needed by direct execution. They note the applicability of their device to prior art systems such as the system illustrated in Figure 3, in which code stored in flash memory **104** is first copied to volatile memory **102** and then executed directly by processor **100** from volatile memory **102**. They also note in passing (column 5 line 33) that, although the primary intended application of their invention is to directly executable flash memories, their invention could also be used with flash memories that are not directly executable, for example with NAND flash memories.

Brown et al. '739 are silent about how code is copied from flash memory **104** to volatile memory **102**. As best understood, the copying is done by processor **100**.

Kakinuma et al. '349 teach that a flash memory controller **2**, separate from a host computer **1**, is conventionally used to copy data from a flash memory **4** to host computer **1**. In the invention of Kakinuma et al. '349, a similar flash memory controller **2** is used to copy data from flash memories **20** and **21** to host computer **1**. In the context of the system illustrated in Figure 3 of Brown et al. '739, flash memory controller **2** would copy data from a flash memory to volatile memory **102** via bus **108** for direct execution by processor **100**, thereby functioning similarly to logic **42** of the present invention.

The crucial difference between the present invention and this combination of Brown et al. '739 and Kakinuma et al. '349 is best understood in reference to Figure 2. Figure 2 shows that logic **42**, flash memory **14** and volatile memory component (S-RAM) **40** communicate with each other via an internal bus **37**. Logic **42** moves code, that is to be executed by CPU **32**, directly from flash memory **14** to volatile memory component **40** via internal bus **37**. Then CPU **32** accesses the code for execution indirectly, via bus **38** and port **12** in addition to bus **37**. This is quite different from the obvious combination of Brown et al. '739 and Kakinuma et al. '349, which would be to have logic **42** move the code to RAM **34** via bus **38** for execution by CPU **32**. In the context of Kakinuma et al. '349, the present invention would have host computer **1** executing code in buffer memories **22** and **23**. In fact, Kakinuma et al. '349 use buffer memories **22** and **23** only to buffer data between flash memories **20** and **21** and host computer **1**. There is neither a hint nor a suggestion in Kakinuma et al. '349 that a processor of host computer **1** could execute code resident in buffer memories **22** and **23**.

These arguments were presented in response to the Office Action mailed on September 24, 2003. The Examiner now has noted that Brown et al. '739 mention, in column 3 lines 63-65, the possibility of coupling volatile memory **102** and flash memory **104** to processor **100** via separate buses. This actually is irrelevant to the present invention because it says nothing about how volatile memory **102** is coupled to flash memory **104**. As best understood, under such an architecture, volatile memory **102** is coupled to flash memory **104** only via processor **100**. In order to use a separate component such as flash memory controller **2** of Kakinuma et al. '349 to copy code from flash memory **104** to volatile memory **102** for execution by processor **100**, a third bus would have to be introduce to couple volatile memory **102** directly to

flash memory **104**. This is precisely the point of innovation of the present invention, as recited in independent claims 1, 12, 18, 20 and 21.

Now, Kakinuma et al. '349 do in fact teach buses **27** and **28** that directly couple flash memories **20** and **21** to buffer memories **22** and **23**. But under MPEP 706.02(j), there must be some suggestion or motivation in either Brown et al. '739 and Kakinuma et al. 349 or in the knowledge generally available to one ordinarily skilled in the art to combine the teachings of these two references. There is no such suggestion or motivation. As noted above, Kakinuma et al. '349 use buffer memories **22** and **23** only to buffer data between flash memories **20** and **21** and host computer **1**, and not to present code to the processor of host computer **1** for execution. One ordinarily skilled in the art would expect host computer **1** to have an architecture similar to the prior art architectures illustrated in Figures 3-5 of Brown et al. '739, with volatile memories **102** for presenting code to processors **100** for execution. There is neither a hint nor a suggestion in Brown et al. '739 and Kakinuma et al. '349, taken separately or in combination, of using memories other than the volatile memories already present in host computer **1** to present code to the processor of host computer **1** for execution. Any assertion to the contrary constitutes impermissible hindsight on the part of the Examiner.

The Examiner also has cited Garner '482 as teaching in column 2 lines 30-34 that code can be transferred from a flash memory to a RAM for execution. This, however, adds nothing to the teachings of Brown et al. '739 in column 3 line 61 through column 4 line 10 regarding copying code from flash memory **104** to volatile memory **102** for execution by processor **100**.

With independent claims 1, 12, 20 and 21 allowable in their present form, it follows that claims 10, 11, 13, 16, 17, 29 and 30, that depend therefrom, also are allowable.

<u>§ 103(a) Rejections – Brown et al. '739 and Kakinuma et al. '349 in view of Garner '482 and Anderson et al. '577</u>

The Examiner has rejected claims 3-5 under § 103(a) as being unpatentable over Brown et al. '739 and Kakinuma et al. '349 and further in view of Garner '482 and Anderson et al., US Patent No. 6,295,577. The Examiner's rejection is respectfully traversed.

It is demonstrated above that independent claim 1 is allowable in its present form. It follows that claims 3-5, that depend therefrom, also are allowable.

<u>§ 103(a) Rejections – Brown et al. '739 and Kakinuma et al. '349 in view of Garner '482 and Mills et al. '688</u>

The Examiner has rejected claims 6 and 7 under § 103(a) as being unpatentable over Brown et al. '739 and Kakinuma et al. '349 and further in view of Garner '482 and Mills et al., US Patent No. 6,385,688. The Examiner's rejection is respectfully traversed.

It is demonstrated above that independent claim 1 is allowable in its present form. It follows that claims 6 and 7, that depend therefrom, also are allowable.

<u>§ 103(a) Rejections – Brown et al. '739 and Kakinuma et al. '349 in view of Garner '482 and Nakata '101</u>

The Examiner has rejected claims 8, 9, 14 and 15 under § 103(a) as being unpatentable over Brown et al. '739 and Kakinuma et al. '349 and further in view of

Garner '482 and Nakata, US Patent No. 6,523,101. The Examiner's rejection is respectfully traversed.

It is demonstrated above that independent claims 1 and 12 are allowable in their present form. It follows that claims 8, 9, 14 and 15, that depend therefrom, also are allowable.

§ 103(a) Rejections – Garfunkel et al. '404 and Chieng et al. '346

The Examiner has rejected claim 19 under § 103(a) as being unpatentable over Garfunkel et al. '404 and Chieng et al., US Patent No. 6,035,346 (henceforth, "Chieng et al. '346"). The Examiner's rejection is respectfully traversed.

As noted above, flash memory **12** of Garfunkel et al. '404 is executable. Therefore, claim 19, that recites a non-executable flash memory, is not obvious from Garfunkel et al. '404 either alone or in combination with any other reference.

Even if Garfunkel et al. '404 had taught a nonexecutable flash memory for storing boot code, claim 19 would not be obvious from the combination of Garfunkel et al. '404 and Chieng et al. '346. Chieng et al. '346 teach a computer system **500** that includes a PCI device **505** whose PROM **520** can be reprogrammed by a host processor **105**. For this purpose, host processor **105** sends a busy signal to PCI device **505** to put the CPU **510** of PCI device **505** into a wait and hold state, downloads reprogramming code to PROM **520**, initializes a memory signature in the RAM **515** of PCI device **505**, and releases CPU **510** from the wait and hold state. This sequence of signals and code flow is <u>opposite</u> to the sequence recited in claim 19. Claim 19 recites sending a busy signal to a processor, transferring boot code from a flash memory to a volatile memory, removing the busy signal, and executing the boot code by the processor. In the context of Garfunkel et al. '404, this would be sending a busy signal to controller **11**, transferring boot code from flash memory **12** to RAM **13**,

removing the busy signal, and executing the boot code by controller **11**. But the analog, in computer system **500** of Chieng et al. '346, of controller **11** of Garfunkel et al. '404, is not CPU **510** but rather host processor **105**. In computer system **500** of Chieng et al. '346, host processor **105** is the <u>source</u> of the busy signal, not the <u>target</u> of the busy signal. One ordinarily skilled in the art would not be led by a study of Chieng et al. '346 to contemplate sending a busy signal <u>to</u> controller **11** of Garfunkel et al. '404 to allow transferring code to a memory for execution by controller **11**.

To further distinguish the present invention from the combination of Garfunkel et al. '404 and Chieng et al. '346, new claim 36 has been added. Claim 36 adds to claim 19 the limitation that the flash-based unit is separate from the processor. This is in contrast with embedded system **10** of Garfunkel et al. '404, that includes flash memory **12** together with controller **11**. This also is in contrast with PCI device **505** of Chieng et al. '346, that includes CPU **510** (the target of the busy signal) together with PROM **520**. Support for new claim 36 is found in the specification in Figure 2 that shows flash-based unit **36** separate from CPU **32**.

<u>§ 103(a) Rejections – Garfunkel et al. '404 and Lee '645</u>

The Examiner has rejected claims 24-28 under § 103(a) as being unpatentable over Garfunkel et al. '404 and Lee, US Patent No. 6,370,645 (henceforth, "Lee '645"). The Examiner's rejection is respectfully traversed.

As noted above, flash memory **12** of Garfunkel et al. '404 is executable. Therefore, claims 24-28, that recite a non-executable flash memory, are not obvious from Garfunkel et al. '404 either alone or in combination with any other reference.

Even if Garfunkel et al. '404 had taught a nonexecutable flash memory for storing boot code, claims 24-28 would not be obvious from the combination of Garfunkel et al. '404 and Lee '645. Lee '645 teaches a hard disk drive that includes a

microprocessor **10**, a flash ROM **40** and a RAM **42**. As described in column 3 lines 55-60 and in column 4 lines 1-9, flash ROM **40** is just large enough (8 Kbytes) to store boot code for the hard disk drive. As described in column 3 line 50, RAM **42** is considerably larger than this (32 Kbytes). Now, in claims 24-28 it is the volatile memory component, not the flash memory, that is only large enough to store the basic initialization boot code. One ordinarily skilled in the art might be led by a study of Lee '645 to make flash memory **12** of Garfunkel et al. '404 just large enough to store boot code, but not to make RAM **13** of Garfunkel et al. '404 just large enough to store boot code.

§ 103(a) Rejections – Brown et al. '739 and Kakinuma et al. '349 in view of Garfunkel et al. '404

The Examiner has rejected claims 31-35 under § 103(a) as being unpatentable over Brown et al. '739 and Kakinuma et al. '349 and further in view of Garfunkel et al. '404. The Examiner's rejection is respectfully traversed.

Claims 31-35 have been canceled, thereby rendering moot the Examiner's rejection of these claims.

In view of the above amendments and remarks it is respectfully submitted that independent claims 1, 12, 18-22, 24-28 and 31-35, and hence dependent claims 3-11, 13-17, 23, 29, 30 and 36 are in condition for allowance. Prompt notice of allowance is respectfully and earnestly solicited.

Respectfully submitted,

Mark M. Friedman
Attorney for Applicant
Registration No. 33,883

Date: March 17, 2004